CSE 40567 / 60567: Computer Security

Software Security 2
Homework #4 Released
Due: 3/2/17 at 11:59PM Eastern Time

See Assignments Page on the course website for details
Mid-term: In-Class March 9th

Have questions? Bring them on Thursday.
Mid-term Course Instructor Feedback (CIF)

10 Points of Extra Credit!
Return to Gabe by next Tuesday

How'm I doin'?
Rainbow table chain

- The tables are made up of chains of hash and reduction functions
- A table only stores the starting plaintext and final hash

After generating many chains the table might look something like:

- iaísudhiú = 4259cc34599c530b1e4a8f225d665802
- oxcvioiux = c744b1716cbf8d4dd0ff4ce31a177151
- 9da8dasf = 3cd696a8571a843cda453a229d741843
- 9da8dasf = 3cd696a8571a843cda453a229d741843
- sodifo8sf = 7ad7d6fa6bb4fd28ab98b3dd33261e8f
Rainbow table algorithm

- Assume Mallory has a hash with an unknown plaintext
- She checks to see whether it is inside any of the generated chains

while(hash not found)
  1. Look for the hash in the list of final hashes, if it is there break out of the loop.
  2. If it isn't there reduce the hash into another plaintext, and hash the new plaintext.
  3. Goto the start.
  4. If the hash matches one of the final hashes, the chain for which the hash matches the final hash contains the original hash.

Mallory can now get that chain's starting plaintext, and start hashing and reducing it, until she comes to the known hash along with its secret plaintext.
Checking the chains

Check the last column of the table: reduce and hash once

Check the second to last column of the table: reduce and hash twice
Checking the chains

Check the third to last column of the table: reduce and hash three times

Match: The starting plaintext (stored with the ending hash) is reduced and hashed until the correct plaintext is found within the chain
One problem: collisions

Two plaintexts hash to the same value:

No guarantee that there will be a hash of a plaintext that will reduce to some other given plaintext.
Solution: use a different reduction function in each column

- Origin of the name “Rainbow Table”
  ‣ If each reduction function is a different color, with starting plaintexts at the top and final hashes at the bottom, the table would look like a rainbow

- Chain merges become rare, because collisions have to occur on the same column (chance of collision is $1 / \text{chain length}$)

- Loops are also solved: if a hash in a chain is the same as a previous hash it won't reduce to the same plaintext.
Color-coded reduction functions

http://www.thesecurityblogger.com/understanding-rainbow-tables/
Password Cracking Tools

Ophcrack (http://ophcrack.sourceforge.net/)
Windows password cracker based on rainbow tables

- Runs on Windows, Linux/Unix, Mac OS X, ...
- Cracks LM and NTLM hashes
- Free tables available for Windows XP and Vista/7
- Live image available to simplify the cracking

Screenshot of Ophcrack version 3.2.0 BY-SA 3.0 Ysangkok
Password Salts

• With a dictionary, it's possible to pre-compute a hash for every word, for all known algorithms

• How do we defend against this?

Calculate a different hash:

\[ H'(\text{username, site, password}) \]

and use the high-order 64 bits as the salt and the low-order 18-24 bits as the iteration count.

Iterations slow down attacks, e.g., if every password is hashed 100,000 times, guessing is slowed down to 1/100,000 the previous rate. (Slows down legitimate use as well.)
OS Authentication
Users

- A user is an identity tied to a single entity
- Specific systems may add additional constraints
- Systems represent user identity in a number of different ways
- The same system may use different representation of identity in different contexts
Good design principles for access control

1. Simplicity makes designs and mechanisms easy to understand

2. Simplicity reduces the potential for inconsistencies within a policy or set of policies

3. Restriction minimizes the power of an entity

4. Entities can communicate with other entities only when necessary

5. “Communication” is used in the widest possible sense, including that of imparting information by not communicating
Principle of Least Privilege

The *principle of least privilege* states that a subject should be given only those privileges that it needs in order to complete its task.

Example: 
userland process
Principle of Separation of Privilege

The principle of separation of privilege states that a system should not grant permission based on a single condition.

**Example:** Ubuntu Linux privilege escalation; user must be in group sudo to use `sudo(8)`

Entry in `/etc/group`: `sudo:x:27:walter`

```
walter@eve:~$ id
uid=1000(walter) gid=1000(walter)
groups=1000(walter),4(adm),24(cdrom),27(sudo)
```
Principle of Least Common Mechanism

The principle of least common mechanism states that mechanisms used to access resources should not be shared.

Example: remote mobile virtualization
Basic OS Authentication

1. Login Prompt
2. Credential
3. Session Granted / Rejected

User

Alice

Has:
Password
One-time password
Biometric

Server

Bob

Checks:
User list
Credential list
ID list
How we log in — in practice

Local (Unity):

Remote (ssh):

$ ssh walter@140.247.178.194
walter@140.247.178.194's password: xxxxxxxxx
Welcome to Ubuntu 12.04.5 LTS (GNU/Linux 3.2.0-88-generic x86_64)
Privilege Levels in Linux

Users have a corresponding ASCII username

`useradd(8)` limits this to 32 characters in Ubuntu

The operating system identifies users by an integer known as a UID

Not all UIDs are created equally:

- root (0), the superuser
- daemon (1) and sys (3), handle some aspects of the network
- lp (7), used for the printer system.
- mail (8), email delivery
- nobody (65534), owns no files and sometimes used as a default user for unprivileged operations
How is privilege enforced?

- **OS System Calls**
- **Linux**: `getuid()` returns the real user id of a calling process
- **Example**: attempt to kill a root owned process as a user

Target process:

```
root     32661  0.0  0.0  21856   380 ?        S    Jan18   0:00 /sbin/udevd --daemon
```

User attempts to kill it:

```
walter@eve:~$ kill -9 32661
-bash: kill: (32661) - Operation not permitted
```

`kill(2)` checks the UID of the calling process and bails out:

```
 getpid()                  = 2391
 kill(32661, SIGKILL)     = -1 EPERM (Operation not permitted)
```
su (1) and sudo (8)

• Logging in as root is considered to be dangerous these days
  ‣ Block root ssh access via `/etc/ssh/sshd_config`
    `PermitRootLogin no` option

• An alternative: `su` to root

• A better alternative: only run one privileged command at a time
  • example: `sudo service apache2 restart`
Groups

• We saw that being in a special group facilitates sudo access

• Users often need to share resources
  ‣ Systems facilitate this by putting users into groups
  ‣ A group is an alias for a set of users
  ‣ Two models: (1) users are assigned groups for the duration of their login session; (2) users can change from one group to another in the same session

Common group:

uid=1000(walter) gid=1000(walter) groups=1000(walter),4(adm),27(sudo),250(cvrl)

uid=1001(gabe) gid=1001(gabe) groups=1001(gabe),250(cvrl)
Linux auth model

Authentication looks simple when you do it, but things are more complicated behind the scenes.

How user information is stored on the local system:

/etc/passwd — All user login information except for the passwords

/etc/shadow — The encrypted passwords
/etc/passwd

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
walter:x:1000:1000:Walter Scheirer,,,:/home/walter:/bin/bash
sshd:x:115:65534::/var/run/sshd:/usr/sbin/nologin
dustin:x:1001:1002:::/home/dustin:/bin/bash
/etc/shadow

root:!:15649:0:99999:7:::
daemon:*:15575:0:99999:7:::
bin:*:15575:0:99999:7:::
sys:*:15575:0:99999:7:::
sync:*:15575:0:99999:7:::
games:*:15575:0:99999:7:::
man:*:15575:0:99999:7:::
walter:$6$yKb3N.3f$qC0X9KVHWV6F/
    T8b.aYJLXdF4hwp9MtMe/
    fLxtCPArX.CgPMRdFw3qwZ1LamMLTpa.p0WplXqGfkPyDrGqBsM
    .:15649:0:99999:7:::
sshd:*:15649:0:99999:7:::
dustin:$6$A2AskP/c$vDB3eqaLNH9wwyn/
hhGjkWt5Wb3uSg6RUgmr3RyqGuscg6ampymUNftU/
PvGwN4tNorlUDmzHJv1qwEK.m5c01:16237:0:99999:7:::

$6$salt$encrypted format indicates SHA-512
Lightweight directory access protocol (LDAP)

Centralized network authentication simplifies user management:
LDAP Protocol

RFC 4511

- Protocol for accessing X.500-based directory services
- Design to run over TCP/IP networks
- LDAP entry is a collection of attributes that has a globally-unique Distinguished Name (DN)
  - \texttt{cn} for common name (Walter Scheirer)
  - \texttt{mail} for email address (walter.scheirer@nd.edu)
- Entries are arranged in a hierarchical tree-like structure
  - Useful for arranging user records into an organization’s structure